This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

Claims 1-33 (Cancelled).

Claim 34 (Currently Amended): A computer-readable medium having computer-executable instructions for performing real-time execution-thread switching comprising:

issuing a first non-maskable interrupt from a counter to an interrupt controller when the counter turns over;

in response to receiving the first non-maskable interrupt, issuing a second non-maskable interrupt from the interrupt controller to a central processing unit;

in an interrupt service routine that services the second non-maskable interrupt,

saving a first execution thread's current state information, wherein the first execution thread is an application-level-code execution thread that does not execute in a most-privileged CPU mode,

setting the counter to specify when the counter will turn over again,

restoring previously stored state information pertaining to a second execution thread, wherein the second execution thread is an application-level-code execution thread that does not execute in a most-privileged CPU mode; and

after execution of the interrupt service routine has finished, executing the second execution thread such that the interrupt service routine that services the second non-maskable interrupt minimizes overhead associated with switching thread execution from the first thread to the second thread.

Claim 35 (Previously Presented): The computer-readable medium of claim 34, wherein the counter is an advanced programmable interrupt controller.

Claim 36 (Previously Presented): The computer-readable medium of claim 34, wherein the first execution thread's current state information includes stack data, processor data, and floating point-unit data.

Claim 37 (Previously Presented): The computer-readable medium of claim 34, wherein the previously stored state information pertaining to a second execution thread includes stack data, processor data, and floating point-unit data.

Claim 38 (Previously Presented): The computer-readable medium of claim 34, wherein the second execution thread is executed after interrupts, which were pending when the interrupt service routine finished, have been executed and after deferred procedure calls, which were pending when the interrupt service routine finished executing, have been executed.

Claim 39 (New): A method for performing real-time execution-thread switching comprising:

issuing a first non-maskable interrupt from a counter to an interrupt controller when the counter turns over;

in response to receiving the first non-maskable interrupt, issuing a second non-maskable interrupt from the interrupt controller to a central processing unit;

in an interrupt service routine that services the second non-maskable interrupt,

saving a first execution thread's current state information, wherein the first execution thread is an application-level-code execution thread that does not execute in a most-privileged CPU mode,

setting the counter to specify when the counter will turn over again,

restoring previously stored state information pertaining to a second execution thread, wherein the second execution thread is an application-level-code execution thread that does not execute in a most-privileged CPU mode; and

after execution of the interrupt service routine has finished, executing the second execution thread such that the interrupt service routine that services the second non-maskable interrupt minimizes overhead associated with switching thread execution from the first thread to the second thread.

Claim 40 (New): The method of claim 39, wherein the counter is an advanced programmable interrupt controller.

Claim 41 (New): The method of claim 39, wherein the first execution thread's current state information includes stack data, processor data, and floating point-unit data.

Claim 42 (New): The method of claim 39, wherein the previously stored state information pertaining to a second execution thread includes stack data, processor data, and floating point-unit data.

Claim 43 (New): The method of claim 39, wherein the second execution thread is executed after interrupts, which were pending when the interrupt service routine finished, have been executed and after deferred procedure calls, which were pending when the interrupt service routine finished executing, have been executed.

Claim 44 (New): A system for performing real-time execution-thread switching comprising:

means for issuing a first non-maskable interrupt from a counter to an interrupt controller when the counter turns over;

means for issuing, in response to receiving the first non-maskable interrupt, a second non-maskable interrupt from the interrupt controller to a central processing unit;

interrupt-service-routine means for servicing the second non-maskable interrupt, including

means for saving a first execution thread's current state information, wherein the first execution thread is an application-level-code execution thread that does not execute in a most-privileged CPU mode,

means for setting the counter to specify when the counter will turn over again,

means for restoring previously stored state information pertaining to a second execution thread, wherein the second execution thread is an application-level-code execution thread that does not execute in a most-privileged CPU mode; and

means for executing the second execution thread, after the interrupt-service-routine means services the second non-maskable interrupt, such that the interrupt-service-routine means minimizes overhead associated with switching thread execution from the first thread to the second thread.

Claim 45 (New): The system of claim 44, wherein the counter is an advanced programmable interrupt controller.

Claim 46 (New): The system of claim 44, wherein the first execution thread's current state information includes stack data, processor data, and floating point-unit data.

Claim 47 (New): The system of claim 44, wherein the previously stored state information pertaining to a second execution thread includes stack data, processor data, and floating point-unit data.

Claim 48 (New): The system of claim 44, wherein the second execution thread is executed after interrupts, which were pending when the interrupt-service-routine means finished servicing the second non-maskable interrupt, have been executed and after deferred procedure calls, which were pending when the interrupt-service-routine means finished servicing the second non-maskable interrupt, have been executed.